

# IoT-Based Machine Learning System for Physical Intrusion Detection Using Dynamic Temperature and Humidity Observations With Raspberry Pi

Kemal Hajdarevic <sup>1</sup>

<sup>1</sup> Faculty of Electrical Engineering, University of Sarajevo, Bosnia and Herzegovina

**Abstract** – This paper presents a system that is able to detect physical intrusion in a specific space based on temperature and humidity change. This specific space was housing hardware components important for information security management infrastructure. Presented system is able to predict that two spaces are connected and that there is a physical breach in protected space. The presented prediction approach involves identifying patterns in historical data, where the subsequent outcomes are already known in advance, and validating these patterns using more recent data. System is implemented using k-Nearest Neighbours, Random Forest, and Support Vector Machine algorithms in Python programming language on Raspberry Pi. Real observed data to predict if specific temperature and humidity indicates intrusion were used. This approach can be used to detect intrusions in the room or in other closed space. More specifically thermal equilibrium phenomenon between two spaces after barrier between them are opened was monitored. Through process of supervised learning using labelled data, system was able to detect intrusion by using k-nearest neighbours, random forest, and support vector machine with different accuracy.

Presented model shows better results using k-nearest neighbours and support vector machine with accuracy of 100% compared to random forest with accuracy of 95%. The system is low cost because of cheap Raspberry Pi controller and sensors.

**Keywords** – KNN, random forest, support vector machine, machine learning, Raspberry Pi.

## 1. Introduction

Information assets based on the ISO 27002 [1] standard are hardware, software, information, infrastructure, people, and services. Focuses of the ISO 27002 standard is preservation of information security i.e. confidentiality, integrity, and availability of information and information assets.

Scientists so far recognise temperature as important factor to predict its changes [2], [3], [4] for managing information security issues in cyber-physical systems. Other usages are also applicable for monitoring environment temperature such as human wellbeing [5].

The relationship between temperature change and information security risk is not straightforward but there are legitimate risks associated with temperature and humidity changes. Hardware components are vulnerable to temperature and humidity extremes. Extreme temperatures can impact the physical infrastructure of data centres.

Beside this, temperature and humidity change can be used as indicator that specific space is suddenly opened. This indicator with other indicators can be used as multilayer physical security of important information security assets such hardware components in cyber-physical environments. Sensors can collect different data type. Collected data holds limited value unless it is thoroughly analysed and used for detections and predictions of events.

---

DOI: 10.18421/TEM132-08

<https://doi.org/10.18421/TEM132-08>

**Corresponding author:** Kemal Hajdarevic,  
Faculty of Electrical Engineering, University of Sarajevo,  
Bosnia and Herzegovina


**Email:** [khajdarevic@etf.unsa.ba](mailto:khajdarevic@etf.unsa.ba)

Received: 13 January 2024.

Revised: 25 March 2024.

Accepted: 08 April 2024.

Published: 28 May 2024.

 © 2024 Kemal Hajdarevic; published by UIKTEN. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 License.

The article is published with Open Access at <https://www.temjournal.com/>

## 2. Research Goals for System Delivery

Goals for this specific system were to choose simplest, fastest, accurate and reliable algorithm to detect and predict intrusion in specific physical space based on temperature and humidity change.

## 3. Related Work and Rationales for System Setup

For this work three machine learning algorithms were used: k-nearest neighbors (k-NN), random forest (RF), and support vector machine (SVM) as algorithms that might be used for experimental system because they were used in similar projects referenced below.

### 3.1. Rationale to Temperature and Humidity Change as an Indicator

The main hypothesis was to simulate thermal equilibrium process between two spaces separated with closed doors after they are opened. It is obvious process that humidity and temperature equilibrium will be started between two closed spaces once they are physically connected.

That fact was used to make detection of that event using hardware and software setup presented in this paper.

### 3.2. Rationale for Deciding to Use KNN, SVM and RF with Real Hardware and Real Data

The k-nearest neighbours (k-NN) [6], [7], [8] classification algorithm utilizes the Euclidean distance metric between two points to assess the proximity of unknown samples to those with known classes. Subsequently, the unknown sample is associated with the most prevalent class within its set of k-nearest neighbours. Based on findings [4] temperature and humidity prediction was determined with accuracy close to 100%.

Based on findings [6] k-NN method accuracy rate was 99.0657% for observed temperature and humidity datasets. In the paper [8] authors employed a machine learning approach, specifically the random forest method, to develop a prediction model for estimating relative humidity.

The study focuses on the accurate modelling of relative humidity using the random forest algorithm, showcasing its application in the context of energy-related research. In the paper [9] authors explored the application of support vector machine (SVM) and Internet of Things (IoT) technologies to predict temperature and humidity levels with potential implications for smart building management and environmental monitoring.

### 3.3. Rationale to Use Real Observed Data and Hardware and Software Components

It was decided to use real hardware and software instead of using simulation environments because it gives better system credibility. An approach of using real hardware and software components was used by other scientists using Raspberry Pi and HT11, HT22 sensors [11], [12].

Raspberry Pi is much cheaper solution than using PC and other more complex system for similar sensor readings [12].

## 4. Hardware and Software Used and Experiment Configuration

For the hardware platform Raspberry Pi 1 B, and two HT11 temperature and humidity sensors were used. For the experimenting purposes environment was set up with which is possible to collect data from two HT11 sensors shown in Figure 1.

Connection scheme between two HT11 sensors and Raspberry Pi 1 GPIO pinouts.

### 4.1. Hardware Used

The Raspberry Pi 1 B, two HT11 temperature, and humidity sensors were used. HT11 sensors were using power supply (+5V) and ground from Raspberry Pi GPIO pinouts. Two GPIO pins (4 and 10) were used for reading data from HT11 sensors.

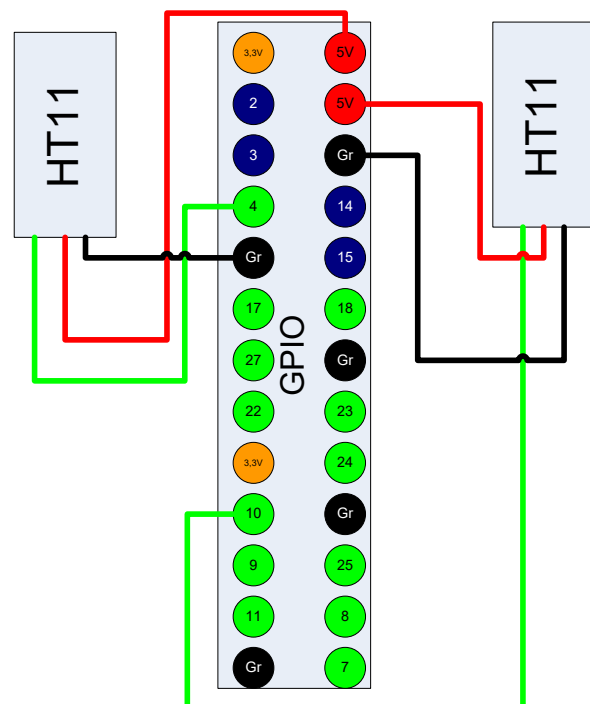


Figure 1. Connection scheme between two HT11 sensors and Raspberry Pi 1 GPIO pinouts

#### 4.2. Software Used

The experimental setup involved the utilization of the Raspbian operating system, tailored specifically for Raspberry Pi hardware, because to its compatibility and optimization for embedded systems. Python, a versatile programming language popular for its ease of use and extensive libraries, served as the primary tool for implementing algorithms and data processing tasks.

The experiment setup relied on several Python libraries, namely Adafruit for hardware interfacing, Pandas for data manipulation, Scikit-learn for machine learning algorithms, Matplotlib and Seaborn for data visualization, and NumPy for numerical computing.

### 5. Experiment Setup and System Design

Raspberry Pi with latest image from official Raspberry Pi and latest distribution of Python were used. Python adafruit, pandas, sklearn matplotlib, and seaborn libraries were used. These libraries provide support for usage of HT11 sensors, KNN, RF, and SVM algorithms and for creating graphics.

Then Python software program was created which is able to read created data model, make reading from attached sensors and printout results and graphics related to KNN, RP, and SVM algorithms. Algorithms allow making multiple classes of data so that system predict to witch class of given data set specific new event belongs especially one that indicates space intrusion.

#### 5.1. Hardware Design

The schematic connections between two HT11 sensors and the GPIO pinouts of Raspberry Pi 1 were depicted in Figure 1. This figure illustrates the connection scheme utilized in this paper.

#### 5.2. Algorithm and Software Design

Firstly Raspbian operating system was installed on Raspberry Pi, than latest Python distribution was installed.

Additionally adafruit, pandas, sklearn matplotlib and seaborn, numpy, libraries were installed. The KNN, RF, SVM are used for detecting and predicting events defined in data model Figure 3 part of data set file – data model.

#### 5.3. K-Nearest Neighbors (K-NN)

The K-NN algorithm [9], [10] relies solely on the provided dataset and a constant parameter, denoted as K. The subsequent steps elucidate the workings of the K-NN algorithm:

- Utilizing a distance function, the K readings closest to the prediction point (next interval) are retrieved.
- The predicted output is calculated as the average of the K nearest readings, as described in the following equation:

$$Y_{i+1} = \frac{1}{K} \sum_{i=1}^K y_i \quad (1)$$

where,

- K is the neighborhood size.
- $y_i$  is the nearest reading.

#### 5.4. Random Forest

The overall objective of a random forest can be seen as:

$$F(x) = \frac{1}{T} \sum_{i=1}^T f_i(x) \quad (2)$$

- $F(x)$  is the final ensemble prediction.
- $T$  is the number of trees in the forest.
- $f_i(x)$  is the prediction of the  $i$ -th tree.

#### 5.5. Support Vector Machine (SVM)

The equation of a hyperplane in a D-dimensional space can be represented as:

$$f(x) = \mathbf{w} \cdot \mathbf{x} + b \quad (3)$$

where:

- $f(x)$  is the decision function.
- $\mathbf{w}$  is the weight vector.
- $\mathbf{x}$  is the input feature vector.
- $b$  is the bias term.

SVM introduces the concept of a hinge loss to penalize misclassifications. The objective function to be minimized is:

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b)) \quad (4)$$

where:

- $N$  is the number of training samples.
- $C$  is the regularization parameter that balances margin maximization and loss minimization.
- $y_i$  is the class label of the  $i$ -th sample.

### 5.6. Data Model Accuracy

Accuracy of algorithms was calculated as follows:

$$\text{Accuracy} = \frac{TP}{N} \quad (5)$$

where:

- TP is the number of true positive correctly classified instances where the actual class is equal to the predicted class,
- N is the total number of instances as the total number of instances in the testing set.

### 5.7. Software Design

To perform data analysis, proposed data flow system for data collection and analysis was used.

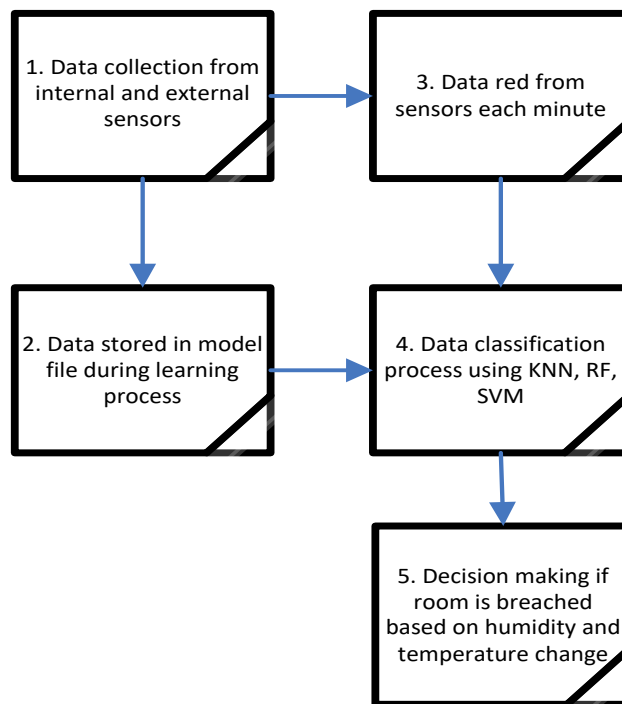


Figure 2. System for Analyzing Temperature and Humidity

The SVM uses a linear boundary between classes a linear kernel (kernel='linear') and a regularization parameter C set to 1.0. Main program set up pin configurations for DHT11 sensors and reads data from sensors. After that program creates a data frame with the new readings acquired from sensors and load the modelled data into memory from the CSV file. For the purpose of predicting the class of the new readings program train the KNN, RF, and SVM models inform and call prediction function to classify new readings.

In Figure 2. a System for Analyzing Temperature and Humidity was depicted where main component and its interactions are presented. First, program initializes libraries needed for reading sensor data. Program has three functions with goal to read data from DHT11 sensors, to train the KNN, RF, SVM models and return the model accuracy, to predict the class using the trained KNN RF, and SVM. KNN uses test\_size of 0.2 which means that 20% of the data will be used for testing, and the remaining 80% will be used for training. The random\_state was 42 which sets the seed for the random number generator. It ensures that the data split is reproducible. Random forest classification in each cycle builds separate data for training and testing random forest model, train model, results prediction for test data set, model evaluation, and detailed results presentation. RF is set up to use 100 trees and ensures reproducibility by setting the random seed to 42.

### 6. Learning Process

For the experiment purposes supervised learning with real data recorded from real sensors readings was used.

```

53,24,71,2,NO_Intrusion
53,24,72,2,NO_Intrusion
52,24,73,2,NO_Intrusion
51,24,73,3,NO_Intrusion
48,25,73,5,NO_Intrusion
48,25,71,10,NO_Intrusion
43,25,55,20,NO_Intrusion
45,25,53,13,NO_Intrusion
39,25,48,23,Intrusion
37,25,46,23,Intrusion
37,25,47,23,Intrusion
37,25,45,23,Intrusion
36,24,45,23,Intrusion
36,23,46,22,Intrusion
    
```

Figure 3. Part of data set file – data model

Data in the data set were labeled in order to classify new data when they are read from sensors using labels: Inside Humidity, Inside Temperature, Outside Humidity, Outside Temperature with data shown in Figure 3 part of data set file – data model.

## 7. Experiment Results

For the experiment purposes system was started in continuous sensor reading mode with goal to watch system behavior.

### 7.1. No Intrusion Detection

First there was no intrusion because two spaces were separated with hard barrier and system did not recognize intrusion.

Current sensor readings:				
Inside Humidity:	50			
Inside Temperature:	25			
Outside Humidity:	51			
Outside Temperature:	11			
KNN Model Accuracy: 1.0				
KNN Class prediction for current sensor readings: NO_Intrusion				
Random Forest Model Accuracy: 0.96				
Classification Report:				
	precision	recall	f1-score	support
Intrusion	0.67	1.00	0.80	2
NO_Intrusion	1.00	0.95	0.98	21
accuracy	0.96			
macro avg	0.83	0.98	0.89	23
weighted avg	0.97	0.96	0.96	23
Random Forest Class prediction for current sensor readings: NO_Intrusion				
SVM Model Accuracy:: 0.9565217391304348				
Classification Report::				
	precision	recall	f1-score	support
Intrusion	0.67	1.00	0.80	2
NO_Intrusion	1.00	0.95	0.98	21
accuracy	0.96			
macro avg	0.83	0.98	0.89	23
weighted avg	0.97	0.96	0.96	23
Matrica konfuzije:				
	[[ 2 0]			
	[ 1 20]]			
SVM Class prediction for current sensor readings:: ['NO_Intrusion']				

Figure 4. System did not recognized Intrusion

System reading and algorithms results are shown in Figure 4 and the results for no intrusion are presented in Figure 5. System as a result shows two clusters: one with blue stars represents no intrusions and one with orange stars represents intrusions.

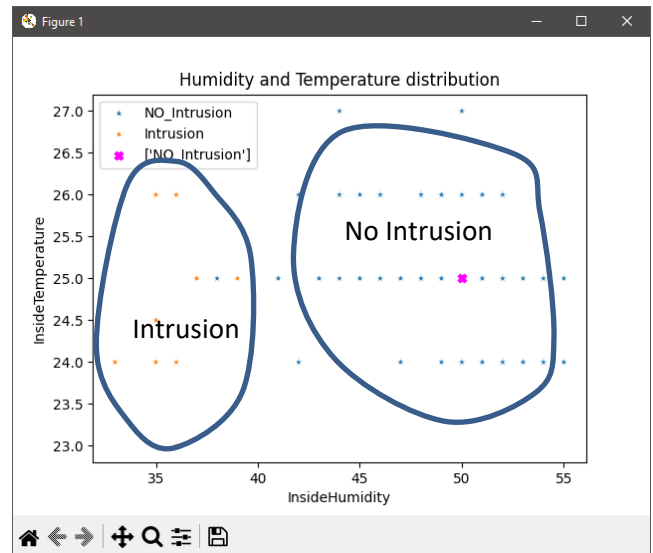


Figure 5. Diagram - results without recognized intrusion event

### 7.2. Intrusion Detection

In experiment phase the doors were opened to make sensors detect readings and to watch system behavior. Experiment shows that system is able to recognize intrusion and it is shown in Figure 6. System recognized intrusion event that x event belongs to cluster with intrusion events shown in diagram Figure 7.

Current sensor readings:				
Inside Humidity:	36			
Inside Temperature:	24			
Outside Humidity:	46			
Outside Temperature:	19			
KNN Model Accuracy: 1.0				
KNN Class prediction for current sensor readings: Intrusion				
Random Forest Model Accuracy: 0.96				
Classification Report:				
	precision	recall	f1-score	support
Intrusion	0.67	1.00	0.80	2
NO_Intrusion	1.00	0.95	0.98	21
accuracy	0.96			
macro avg	0.83	0.98	0.89	23
weighted avg	0.97	0.96	0.96	23
Random Forest Class prediction for current sensor readings: Intrusion				
SVM Model Accuracy:: 0.9565217391304348				
Classification Report::				
	precision	recall	f1-score	support
Intrusion	0.67	1.00	0.80	2
NO_Intrusion	1.00	0.95	0.98	21
accuracy	0.96			
macro avg	0.83	0.98	0.89	23
weighted avg	0.97	0.96	0.96	23
Matrica konfuzije:				
	[[ 2 0]			
	[ 1 20]]			
SVM Class prediction for current sensor readings:: ['Intrusion']				

Figure 6. System recognized intrusion event

System shows that for given data set KNN has 100% accuracy while RF and SVM gives 96%.

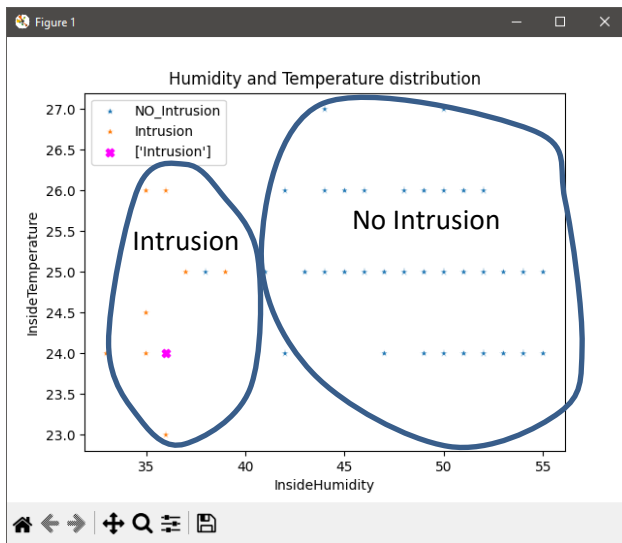


Figure 7. Diagram - recognized intrusion event

Decreased number of trees from 100 to 2 resulted in different classification or false positive for RF.

In Figure 8 comprehensive results for intrusion correctly detected are shown.

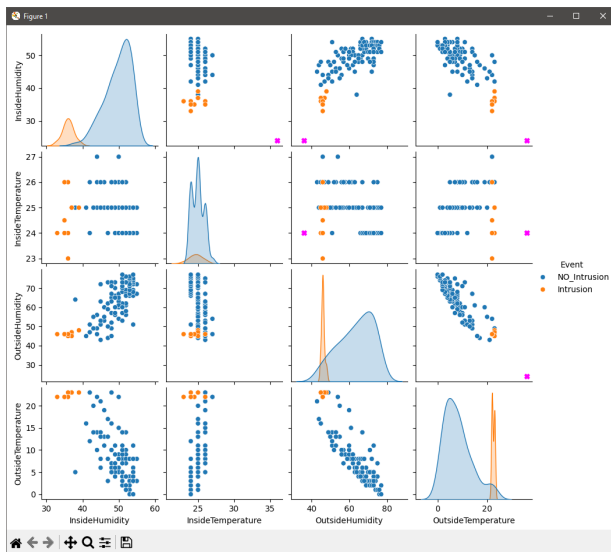


Figure 8. Diagram - comparative results

### 7.3. False Positives

The system was fine-tuned with various parameters, albeit not all, to prevent exceeding the primary objectives of simplicity, speed, and accuracy in algorithm selection. Interestingly, increasing the number of trees to 50 yielded identical results to those obtained with 100 trees for the Random Forest (RF) algorithm. This finding is depicted in Figure 9, where the false positive results for the same reading as presented in Figure 6 are shown.

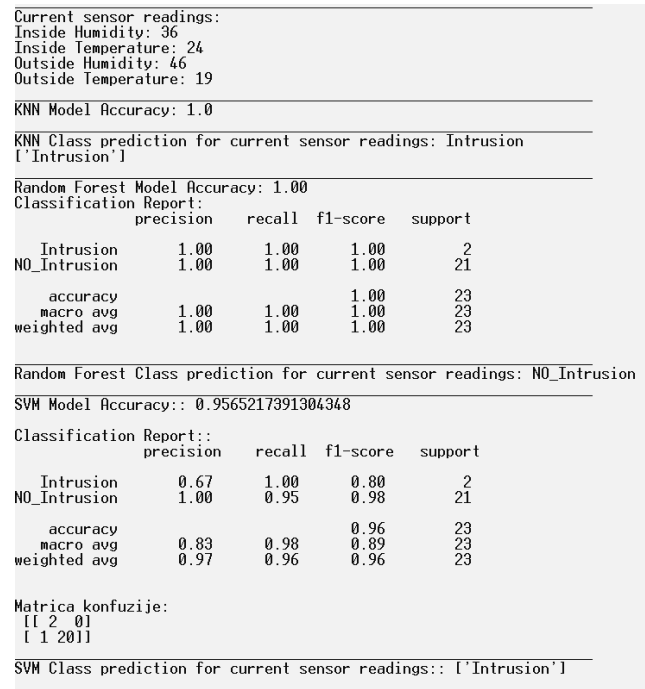


Figure 9. False positive results 1

Less number of trees speeds algorithm but decreasing number of trees cannot capture the underlying patterns effectively, resulting in a model that does not generalize well to unseen data shown in Figure 10.

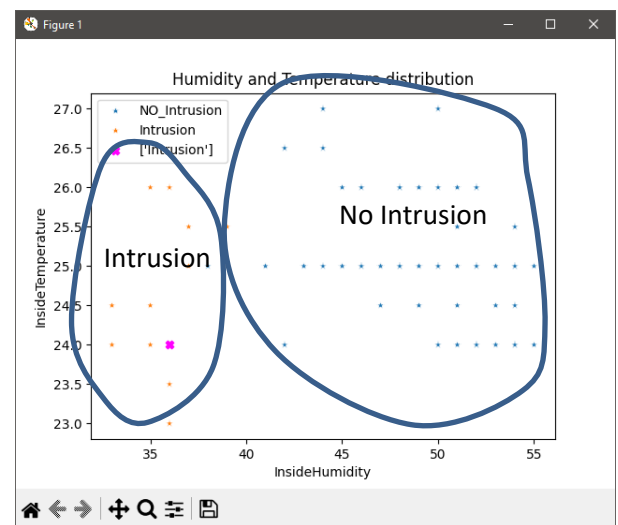


Figure 10. Diagram - false positive results 1

Additional tests were conducted by manually inputting data into the system, mimicking the data obtained from sensors. The KNN detect intrusion and RF and SVM show no intrusion. As shown in Figure 11. and Figure 12 this result shows that KNN depends on data set more than RF and SVM.

```

Current sensor readings:
Inside Humidity: 23
Inside Temperature: 24
Outside Humidity: 46
Outside Temperature: 1
KNN Model Accuracy: 1.00
KNN Class prediction for current sensor readings: NO_Intrusion
['NO_Intrusion']
Random Forest Model Accuracy: 0.96
Classification Report:
      precision    recall  f1-score   support

Intrusion      0.67      1.00      0.80         2
NO_Intrusion   1.00      0.95      0.98        21

   accuracy
macro avg      0.83      0.98      0.89        23
weighted avg   0.97      0.96      0.96        23

Random Forest Class prediction for current sensor readings: Intrusion
SVM Model Accuracy:: 0.9565217391304348
Classification Report::
      precision    recall  f1-score   support

Intrusion      0.67      1.00      0.80         2
NO_Intrusion   1.00      0.95      0.98        21

   accuracy
macro avg      0.83      0.98      0.89        23
weighted avg   0.97      0.96      0.96        23

Matrica konfuzije:
[[ 2  0]
 [ 1 20]]
SVM Class prediction for current sensor readings:: ['Intrusion']
    
```

Figure 11. False positive results 2

All test results were not presented in this paper because of limited space.

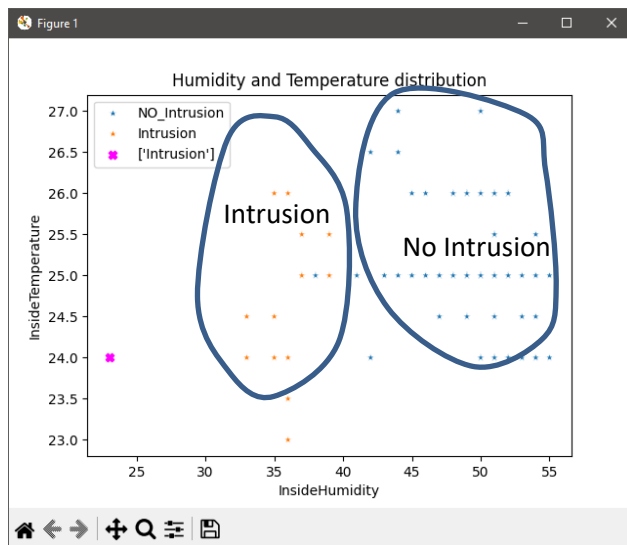


Figure 12. Diagram - false positive results 2

## 8. Conclusions

From presented experiments it can be concluded that cheap hardware Raspberry Pi and HT11 sensors can be used with Python interpreter to build useful system for predicting intrusion events based on temperature and humidity change. Presented system setup with real hardware and data can be used for other applications as well.

### 8.1. KNN Comparison to RF and SVM

KNN shows better results compared to RF and SVM because of possible reasons described below:

**Local Patterns and Non-Linearity:** KNN is a non-parametric algorithm that works well in capturing local patterns and non-linear relationships in the data. **Simplicity of the Model:** KNN is an instance-based learning algorithm that does not build an explicit model during training. Instead, it memorizes the training instances and makes predictions based on the closest neighbors.

**Data Size:** KNN performs well with smaller datasets, particularly when there is enough density in the feature space to make reliable local predictions. SVM and RF require more data to generalize effectively and they might struggle with overfitting in situations with limited data.

**Few Hyper parameters:** KNN has fewer hyper parameters to tune compared to SVM and random forest. The computation time of the three algorithms was not quantified. However, it is worth noting that KNN utilizes fewer parameters for calculation. This is another factor reinforcing reliance solely on the KNN algorithm within the presented system.

System is cheap, easy to setup, configure and more important easy to change configuration setups for algorithms and interchange real sensor readings and manual data inputs.

### References:

- [1]. International Organization for Standardization. (2022). *Information technology — Security techniques — Code of practice for information security controls*. (ISO/IEC 27002:2022). ISO.
- [2]. Jeong, W., Choi, E., Song, H., Cho, M., & Choi, J. W. (2022). Adaptive Controller Area Network Intrusion Detection System Considering Temperature Variations. *IEEE Transactions on Information Forensics and Security*, 17, 3925-3933.
- [3]. Badhiye, S. S., Wakode, B. V., & Chatur, P. N. (2012). Analysis of Temperature and Humidity Data for Future value prediction. *International Journal of Computer Science and Information Technologies*, 3(1), 3012-3014.
- [4]. Labrado, C., Thapliyal, H., Prowell, S., & Kuruganti, T. (2019). Use of thermistor temperature sensors for cyber-physical system security. *Sensors*, 19(18), 3905.
- [5]. Keerthi, A. M., Raksha, R., & Rakesh, N. (2020). A novel remote monitoring smart system for the elderly using internet of things. In *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 596-602. IEEE. Doi: 10.1109/ICECA49313.2020.9297403.
- [6]. Prayitno, E., Fahmi, N., Al Rasyid, M. U. H., & Sudarsono, A. (2021). An implentation of IoT for environmental monitoring and its analysis using k-NN algorithm. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 19(6), 1811-1819.

- [7]. Badhiye, S. S., Sambhe, N. U., & Chatur, P. N. (2013). KNN technique for analysis and prediction of temperature and humidity data. *International Journal of Computer Applications*, 61(14).
- [8]. Qadeer, K., Ahmad, A., Qyyum, M. A., & Lee, M. (2019). Relative humidity estimation: machine learning approach-random forest-based prediction model. *International Conference on Applied Energy*, 12-15.
- [9]. Albert, A., Nuha, H. H., & Mugitama, S. A. (2023). Prediction of Temperature and Humidity at Telkom University Landmark Tower (TULT) Using Support Vector Machine (SVM) and Internet of Things (IoT). In *2023 International Conference on Data Science and Its Applications (ICoDSA)*, 225-229. IEEE.
- [10]. Zhao, H., & Chen, Y. (2023). A Comparative Study for Temperature Prediction by Machine Learning and Deep Learning. In *2023 International Conference on Intelligent Computing and Control (IC&C)*, 77-84. IEEE. Doi: 10.1109/IC-C57619.2023.00020.
- [11]. Saeed, U., Jan, S. U., Lee, Y. D., & Koo, I. (2020). Machine learning-based real-time sensor drift fault detection using Raspberry PI. In *2020 International Conference on Electronics, Information, and Communication (ICEIC)* (pp. 1-7). IEEE.
- [12]. Fowdur, T. P., Beeharry, Y., Hurbungs, V., Bassoo, V., Ramnarain-Seetohul, V., & Lun, E. C. M. (2018). Performance analysis and implementation of an adaptive real-time weather forecasting system. *Internet of Things*, 3, 12-33.